Scraping Company Data

Need to scrape reputable websites:
What is Reputable?

- Start with company sites as its more likely to be more authentic experience
    - Career Pages, company blogs?
- Look into sites that provide more personalized experiences & issues faced (Glassdoor, etc)
    - Look further into other sites such as LinkedIn, Indeed, Blind, reddit posts (how to identify high quality posts is something to look into?) as well as Discord servers for interview preparation & mentoring (Again how do we decide if something can be collected for use is something to look into further?)

Why start with Glassdoor for example?

- Lots of information across so many different roles.
- Users are somewhat more "verified" -> requires an account
- https://scrapfly.io/blog/how-to-scrape-glassdoor/ -> resource to web scrape

Scraping Glassdoor:

1. Remove Overlay -> script
2. Provide region to site -> we can get this from job description
3. Get ID for company -> each company has associated ID -> script
4. pagination scraping algorithm runs for interviews page -> Graphql is being used on glassdoor which we can use for gathering data

How does pagination scraping algorithm work?

- Scrape the first jobs page
- Extract GraphQl cache for jobs data
- Parse HTML for total page count
- Scrape remaining pages concurrently

What is GraphQL?

- GraphQL is a language for APIs that allows clients to request only the data they need.
- Instead of multiple REST API endpoints -> a single GraphQL endpoint can handle complex queries
- Returns structured responses

How will our request to Glassdoor work?

- We will send a POST request to a GraphQL endpoint

Example Request:

```
{
  "operationName": "InterviewList",
```

```
  "variables": {
    "companyId": 12345,
    "jobTitle": "Software Engineer",
    "limit": 10
  },
  "query": "query InterviewList($companyId: Int!, $jobTitle: String!, $limit: Int!) {
interviews(companyId: $companyId, jobTitle: $jobTitle, limit: $limit) { id question answer difficulty
date }}"
}
```

What is happening above?

- We name the operation
- We assign the variables we have / will be using
- Define a query -> will be used to fetch the data & be assigned
- This is what the request contains -> returns a structured JSON

What will be done after structured JSON is returned?

- data can be parsed and stored for analysis.

Why is Graphql good for this job compared to REST?

- Used for glassdoor -> use what is already being used.
- GraphQL allows requesting only the necessary fields, reducing data load
- Easier to clean up excess data -> queries return well-organized JSON objects.

Issues to Keep in mind with glassdoor:

- Glassdoor is known for its high blocking rate.
- Automated scripting is again ToS -> could ban IP or take legal action -> Marawan mentioned we can use a proxy server for this? -> proxy will also help with rate limiting due to limit for requests in short time
- Get around CAPTCHA Bot Detection -> Headless browser? -> rotating proxies, setting real request headers, randomizing request rates could all be solutions aswell as mentioned in this article https://brightdata.com/blog/web-data/how-to-scrape-glassdoor

There is a glassdoor scraping API:

- No rotating proxies need / Handles CAPTCHAS & restrictions / no need for a scraping infrastructure for Glassdoor -> Depending on usage -> Can be very expensive depending on number of reviews scraped (I think less than 50k reviews could be affordable)

Cool now we've scraped our data, now what?

- Store into DB
- Sentiment Analysis?

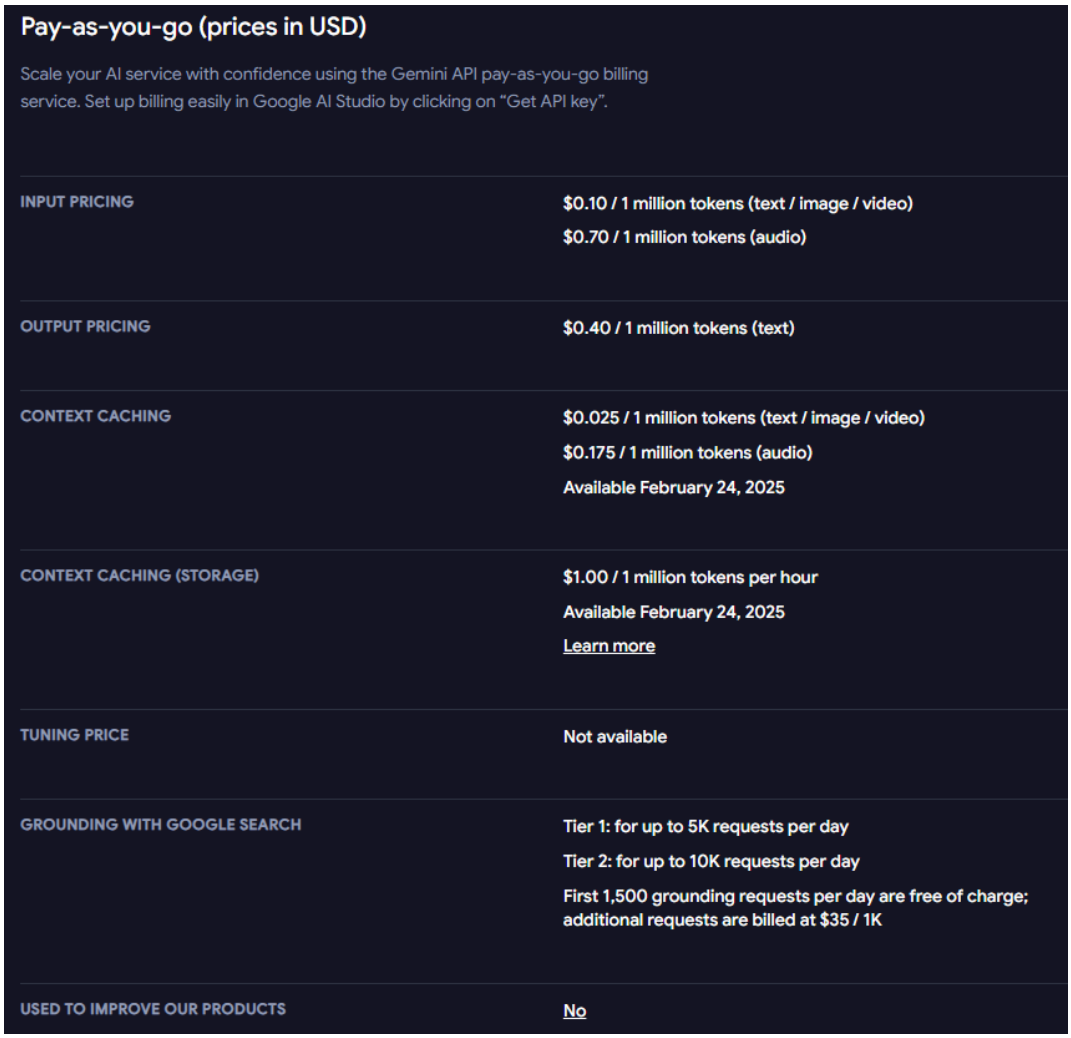
Vectorized DB options:
- Pinecone

Pricing:

- Gemini V2

https://ai.google.dev/pricing#2_0flash

Free tier up to 1 million tokens of storage per hour -> data collected by google



Pay-as-you-go (prices in USD)

Scale your AI service with confidence using the Gemini API pay-as-you-go billing service. Set up billing easily in Google AI Studio by clicking on "Get API key".

INPUT PRICING	$0.10 / 1 million tokens (text / image / video) $0.70 / 1 million tokens (audio)
OUTPUT PRICING	$0.40 / 1 million tokens (text)
CONTEXT CACHING	$0.025 / 1 million tokens (text / image / video) $0.175 / 1 million tokens (audio) Available February 24, 2025
CONTEXT CACHING (STORAGE)	$1.00 / 1 million tokens per hour Available February 24, 2025 Learn more
TUNING PRICE	Not available
GROUNDING WITH GOOGLE SEARCH	Tier 1: for up to 5K requests per day Tier 2: for up to 10K requests per day First 1,500 grounding requests per day are free of charge; additional requests are billed at $35 / 1K
USED TO IMPROVE OUR PRODUCTS	No

- Deepseek

Pricing Details

USD CNY

MODEL[1]	CONTEXT LENGTH	MAX COT TOKENS[2]	MAX OUTPUT TOKENS[3]	1M TOKENS INPUT PRICE (CACHE HIT) [4]	1M TOKENS INPUT PRICE (CACHE MISS)	1M TOKENS OUTPUT PRICE
deepseek-chat	64K	-	8K	$0.07	$0.27	$1.10
deepseek-reasoner	64K	32K	8K	$0.14	$0.55	$2.19 [5]

- (1) **The `deepseek-chat` model has been upgraded to DeepSeek-V3.** `deepseek-reasoner` points to the new model **DeepSeek-R1**.
- (2) **CoT (Chain of Thought)** is the reasoning content `deepseek-reasoner` gives before output the final answer. For details, please refer to Reasoning Model。
- (3) If `max_tokens` is not specified, the default maximum output length is 4K. Please adjust `max_tokens` to support longer outputs.
- (4) Please check DeepSeek Context Caching for the details of Context Caching.
- (5) The output token count of `deepseek-reasoner` includes all tokens from CoT and the final answer, and they are priced equally.

- Openai Whisper:

Transcription and speech generation

Model	Use case	Cost
Whisper	Transcription	$0.006 minute
TTS	Speech generation	$15.00 1M characters
TTS HD	Speech generation	$30.00 1M characters

- Groq esque -> more models?? -> check pricing aswell (llm + service)
- RunPod -> could be good if we wanna host our own model
- HuggingFace Inference API -> most models on here -> probably cheapest aswell

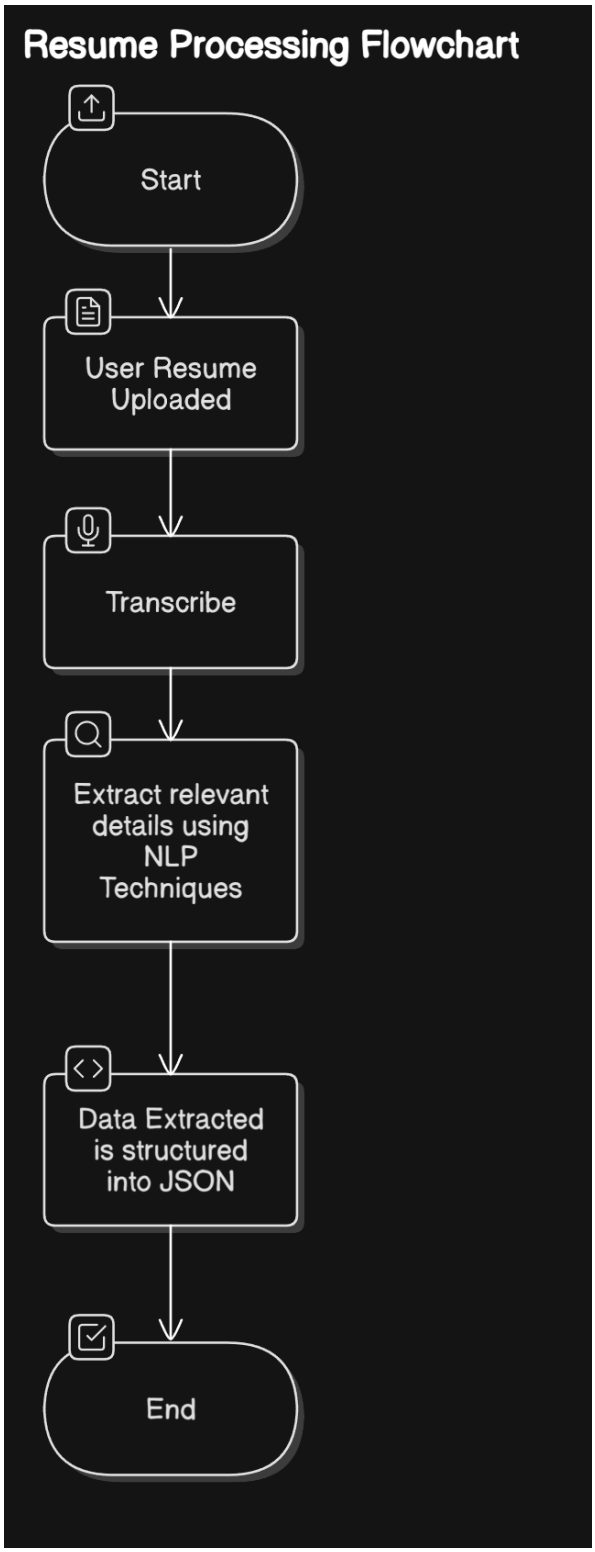
https://huggingface.co/pricing

- https://replicate.com/pricing

Diagram:
Workflow

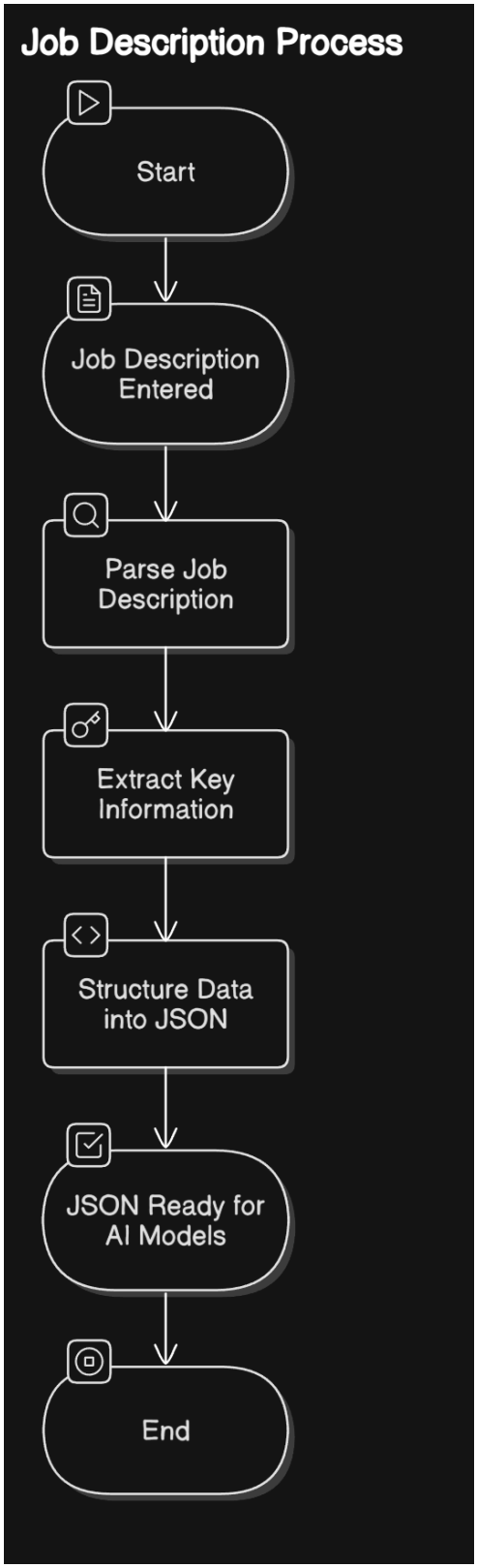
Data Collection & Preprocessing:

Resume Processing Process: User Resume Uploaded -> Transcribe -> Extract relevant details using NLP Techniques



Job Description Process: Job Description Entered -> Parse to find key information (Something similar to this https://www.affinda.com/job-description-parser)

Depending on UI -> define order of execution of processes above



Cleaning & Structuring Process: Data Extracted is structured into JSON for easier use by AI models

What are the relevant details?

Resume:

- Work experiences -> Job title, companies worked at, responsibilities, duration
- Skills -> Any skills listed, certifications, etc
- Education -> Degree, University name, graduation date, relevant coursework
- Projects -> Description, achievements

Job Description:

- Job Title, Responsibilities, Skills required, experience level, Qualifications/degrees, any attributes they want mentioned

What NLP techniques can be used?

- Named Entity Recognition -> Identify & classify job titles, companies, dates, skills, locations etc
- POS (Parts-of-Speech Tagging) -> Helps to understand grammatical structure of the text

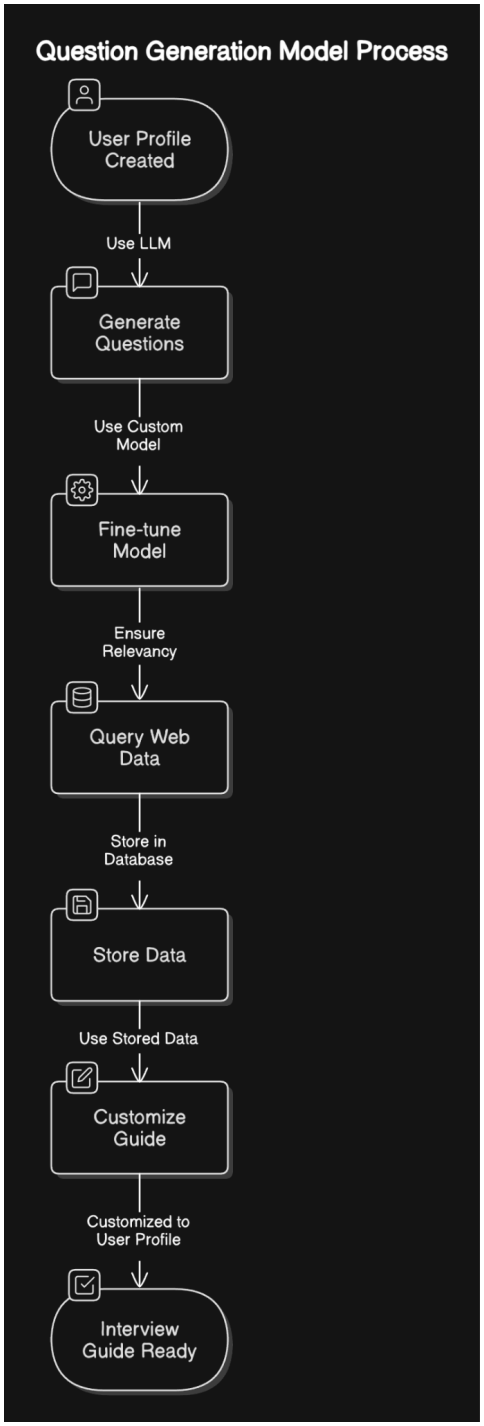
Example service: https://www.affinda.com/job-description-parser
https://stackoverflow.com/questions/54023471/parsing-identifying-sections-in-job-descriptions

Why use JSON for structuring?

- Easier to parse with most programming languages(readable, compact, flexibility)

Question Generation Model Process:

User Profile created -> use an LLM (options above) to generate interview questions / guide to prepare -> Custom fine tuned model can be used to make sure questions are specific to the role & skills that match the user's profile (Use a query to web scraped data & store into database to be used by this model when needed) -> Guide for interview questions customized to the user's profile (role & resume)



Response Analysis Model Process:

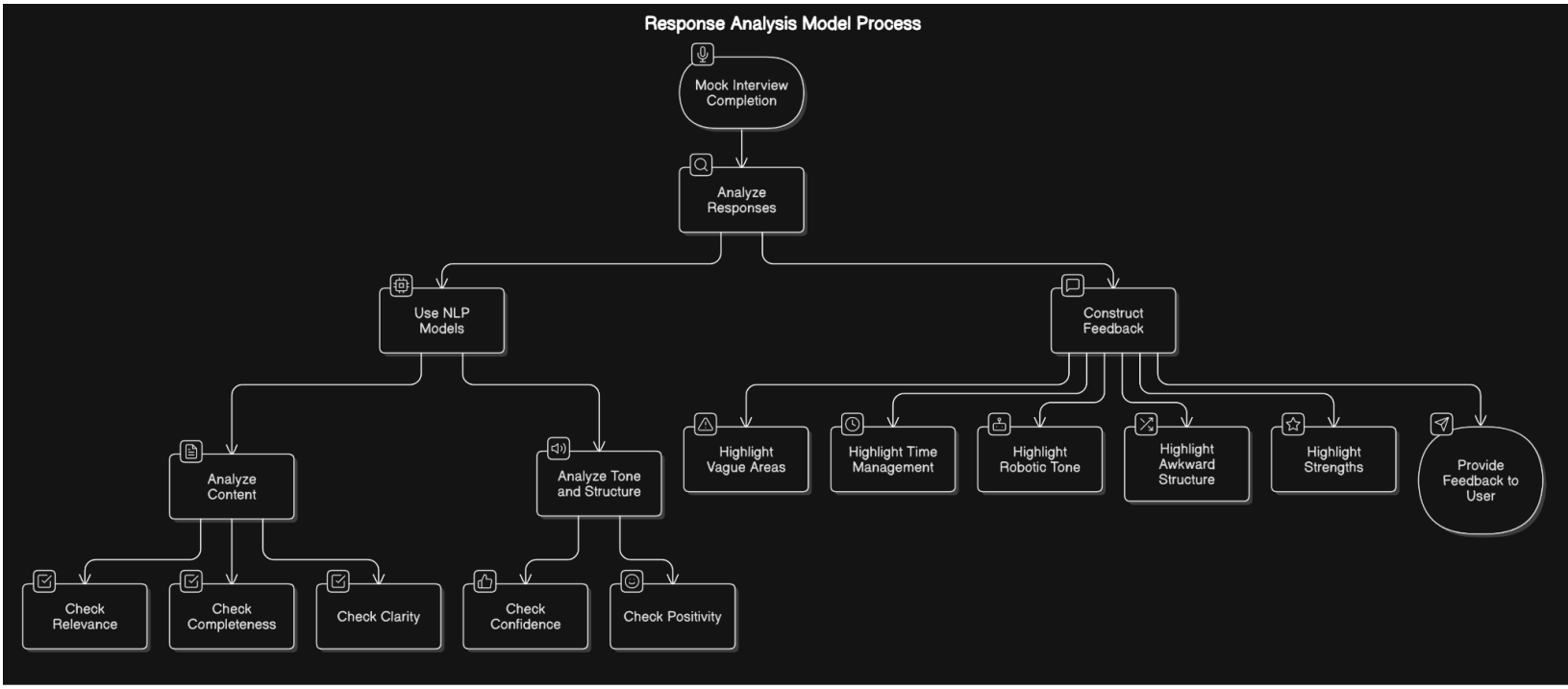
After completion of mock interview voice recordings -> Use NLP models such as Whisper to analyze responses -> Return feedback for user

What is analyzed by the NLP model in this stage?

- Analyze content of the response against webscraped responses -> relevant? Complete? Clear?
- Analyze the tone & structure of the response -> confident? positive?

How is the feedback given to the user?

- Constructive feedback based on the quality as measured by analysis above & relevance of responses -> if something is vague or too much time spent on a certain area -> let them know
- Does it sound too robotic? Is it too awkwardly structured?
- What was done well in your responses to keep doing?



Additional Resources to look at:

- https://github.com/NirDiamant/RAG_Techniques
- https://bugfree.ai/mock https://yoodli.ai/ -> similar services
- https://scrapegraphai.com/ -> scraping tool that could be used
- https://github.com/unclecode/crawl4ai -> web scraping tool -> structures data for llms